Problem J. Zubereitung

Input file:	`zubereitung.in`
Output file:	`zubereitung.out`
Time limit:	2 seconds
Memory limit:	256 mebibytes

Andrew is preparing problems for his next contest in Petrozavodsk. He has two types of problems: easy problems take t_1 minutes to prepare each, and hard problems take t_2 minutes to prepare each ($t_1 < t_2$). Each day Andrew has d minutes to spend on preparing problems.

Andrew has decided on the following algorithm for picking the order in which he prepares the problems. First, he arranges all problems he has to prepare in a fixed sequence. Each day, he goes through this sequence from left to right, and whenever he encounters a problem that is not yet prepared but can be prepared today, he prepares it, then continues looking through the sequence. Note that he does not stop looking if he encounters a problem that he doesn't have enough time to prepare — maybe another problem further down the sequence requires less time. On the next day, he starts looking at the sequence again from the very beginning (of course, skipping the problems that he has already prepared), and so on until all problems are prepared.

Andrew is wondering: is this algorithm optimal? More precisely, do there exist two sequences of problems, which contain the same set of problems (in other words, the same amount of easy problems and the same amount of hard problems), but require different number of days to prepare if Andrew follows his algorithm?

Input

The first line of the input file contains the number of testcases t, $1 \le t \le 1000$. The next t lines contain one testcase each, described by three integers t_1, t_2 and d, $1 \le t_1 < t_2 \le d \le 100$. All testcases in one input file are different.

Output

For each testcase, output two sequences of problems, one per line, that make Andrew's algorithm spend different number of days, but have the same set of problems. Each sequence should be printed without spaces, with character '`E`' denoting an easy problem and character '`H`' denoting a hard problem. The length of each sequence must be at most 10000. It is guaranteed that if a solution exists, there exists a solution where each sequence has length at most 10000. If there's no solution for a particular testcase, print "`OPTIMAL`" (without quotes) on one line.

Examples

zubereitung.in	zubereitung.out
2 1 2 2 1 2 3	OPTIMAL EEEHHH HHHEEE

Note

In the second example case, ordering `EEEHHH` needs four days to prepare: three easy problems in the first day, and one hard problem in each of the following days. At the same time, `HHHEEE` has the same set of problems and requires just 3 days: every day Andrew would start with a hard problem, and then skip over remaining hard problems until the next easy problem, so he would prepare a hard problem and an easy problem each day.